WEDNESDAY 6 OCTOBER 1993

9-15 Overview of SOFTWARE modules (WMB)

9-25 Module S2 (MSJ)

9-45 Module S1 (???)

10-00 Module S3 (MJ)

10-20 Coffee

11-00 Tour of Library

11-45 Module S4 (WMB)

12-45 Lunch (not provided)

2-00 Overview of THEORY modules (AMG)

2-10 Module T1 (AMG)

2-30 Module T2 (MSP)

2-50 Module T3 (DW)

3-10 Question & Answer Session (GRM, AMG, WMB +)

3-30 Finish

~~~~~~~~~~~~~~~~~~~

GRM/28.09.93
~~~~~~~~~~~~~~~~~~~

Software Modules

S1 Concurrent Software Implementation

S2 Concurrency in Programming Languages

S3 Parallelism and Real-Time Systems

S4 Definitive Methods for Concurrent Systems Modelling

Sources for software: concurrent and parallel

- operating systems culture

  many agents trying to use a single resource
  a single processor optimising
  the use of many peripherals

- distributed computing

  many small processors networked
  communication and local processing
  distributed databases
  message passing / information hiding

- parallel computers

  every machine is parallel
  at some level of abstraction?
  programmable in a parallel fashion
  can specify its activity as
  parallel action of many agents
  SIMD, MIMD processors

- machines and applications

  special purpose parallel machines
  systolic arrays

- reactive systems

  specifying concurrent computation involving
  perception, communication, action
  real-time constraints

Issues in software for concurrency / parallelism

- what is a good way to describe parallel activity?
- how should we **analyse** and **reason** about parallel programs?
- can we **avoid** explicit state-based programming models?
- is there a good **general-purpose** model of parallel computation?
- how do we relate real-world requirements to the idiosyncrasies of abstract parallel machines?
- what is the relationship between **modelling** and **programming**?
- is there a theoretical framework for computation that embraces behaviour of concurrent systems?

Issues in software for concurrency / parallelism

To a qualified extent ....

- there are good ways to describe sequential activity
- we can **analyse** and **reason** about sequential programs
- we can **avoid** explicit state-based programming models
- there is a good **general-purpose** model of sequential computation

BUT

concurrency is essentially involved in
- relating real-world requirements to the idiosyncrasies of any abstract machine

sequential programming does not adequately address
- the relationship between **modelling** and **programming**

the classical theory of computation does not provide
- a theoretical framework for computation that embraces behaviour of concurrent systems

SCRIPT

- operating systems culture 1960s DIJKSTRA

  semaphores
  dining philosopher's deadlock, mutual exclusion
  Petri Nets

- distributed computing early 1970s
  transputer culture David May EPL → OCCAM
  unfulfilled expectations re future of hardware?
  applications: airline reservations etc.
  Parnas et al modularisation, distribution.

- parallel computers late 1970s
  synchronisation by dictat
  fixed patterns of organisation of process & communication
  CRAY, DAP, Parallel Fortran, linear algebra
  parallel action in processing, ~~reading~~ input, output
  mouse, keyboard, tactile devices, audio

- machines and applications early 1980s
  can we have general-purpose parallelism?
  special-purpose parallel machines
  systolic arrays
  neural nets

- reactive systems
  aims & tasks
  human + electronic devices +
  + sensors + processors